

RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	FFFFFFFFFFF	FFFFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	FFFFFFFFFFF	FFFFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	FFFFFFFFFFF	FFFFFFFFFFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	FFF	FFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	FFFFFFFFF	FFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	FFFFFFFFF	FFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	FFFFFFFFF	FFFFFFFFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF	
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF	
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF	

```
EEEEEEEEEE NN NN DDDDDDDD CCCCCCCC HH HH RRRRRRRR
EEEEEEEEEE NN NN DDDDDDDD CCCCCCCC HH HH RRRRRRRR
EE NN NN DD DD CC HH HH RR RR
EE NN NN DD DD CC HH HH RR RR
EE NNNN NN DD DD CC HH HH RR RR
EE NNNN NN DD DD CC HH HH RR RR
EEEEEEEE NN NN DD DD CC HHHHHHHHHH RRRRRRRR
EEEEEEEE NN NN DD DD CC HHHHHHHHHH RRRRRRRR
EE NN NNNN DD DD CC HH HH RR RR
EE NN NNNN DD DD CC HH HH RR RR
EE NN NN DD DD CC HH HH RR RR
EEEEEEEE NN NN DDDDDDDD CCCCCCCC HH HH RR RR
EEEEEEEE NN NN DDDDDDDD CCCCCCCC HH HH RR RR
```

```
LL LL SSSSSSSS
LL LL SSSSSSSS
LL II
LL II
LL II
LL II
LL II
LL II
LL II
LL II
LL II
LLLLLLLLLL II II SSSSSSSS
LLLLLLLLLL II II SSSSSSSS
```



```
1 0001 0 %TITLE 'Process character termination.'
2 0002 0 MODULE ENDCHR ( IDENT = 'v04-000'
3 P 0003 0 %BLISS32 [ , ADDRESSING_MODE ( EXTERNAL = LONG_RELATIVE,
4 0004 0 NONEXTERNAL = LONG_RELATIVE) ]
5 0005 0 ) =
6 0006 1 BEGIN
7 0007 1
8 0008 1 *****
9 0009 1 *
10 0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
11 0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
12 0012 1 * ALL RIGHTS RESERVED.
13 0013 1 *
14 0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
15 0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
16 0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
17 0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
18 0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
19 0019 1 * TRANSFERRED.
20 0020 1 *
21 0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
22 0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
23 0023 1 * CORPORATION.
24 0024 1 *
25 0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
26 0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
27 0027 1 *
28 0028 1 *
29 0029 1 *****
30 0030 1
31 0031 1 ++
32 0032 1 FACILITY: DSR (Digital Standard RUNOFF) / DSRPLUS
33 0033 1
34 0034 1 ABSTRACT: Put character and 'emphasis' indicators into the MRA.
35 0035 1
36 0036 1
37 0037 1 ENVIRONMENT: Transportable
38 0038 1
39 0039 1 AUTHOR: R.W.Friday
40 0040 1
41 0041 1 CREATION DATE: May, 1978
42 0042 1
```

ENDCHR
V04-000

Process character termination.
Revision History

M 15
16-Sep-1984 00:23:49
14-Sep-1984 13:06:07

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]ENDCHR.BLI;1

Page 2
(2)

```
.. 44      0043 1 %SBTTL 'Revision History'
.. 45      0044 1
.. 46      0045 1   MODIFIED BY:
.. 47      0046 1
.. 48      0047 1       009      REM00009      Ray Marshall      13-Sep-1983
.. 49      0048 1      Conditionalized some of the logic added in the previous edit
.. 50      0049 1      level so that it will properly build RUNOFF images.
.. 51      0050 1      Modified the way we look at the character output. We strip
.. 52      0051 1      off the high order bit so we can still check in the same
.. 53      0052 1      way if is printable.
.. 54      0053 1
.. 55      0054 1       008      KFA00008      Ken Alden      09-Sep-1983
.. 56      0055 1      For DSRPLUS: Improved functionality for passthrough.
.. 57      0056 1
.. 58      0057 1       007      KFA00007      Ken Alden      16-Mar-1983
.. 59      0058 1      For DSRPLUS: added functionality for passthrough flag.
.. 60      0059 1
.. 61      0060 1       006      KAD00006      Keith Dawson    07-Mar-1983
.. 62      0061 1      Global edit of all modules. Updated module names, idents,
.. 63      0062 1      copyright dates. Changed require files to BLISS library.
.. 64      0063 1
.. 65      0064 1  !--
```



```

67      0065 1 %SBTTL 'Module Level Declarations'
68      0066 1
69      0067 1
70      0068 1 | TABLE OF CONTENTS:
71      0069 1 |
72      0070 1 |
73      0071 1 | INCLUDE FILES:
74      0072 1 |
75      0073 1
76      0074 1 LIBRARY 'NXPORT:XPORT';           ! XPORT Library
77      0075 1 REQUIRE 'REQ:RNODEF';           ! RUNOFF variant definitions
78      0206 1
79      U 0207 1 %IF DSRPLUS %THEN
80      U 0208 1 LIBRARY 'REQ:DPLLIB';           ! DSRPLUS BLISS Library
81      0209 1 %ELSE
82      0210 1 LIBRARY 'REQ:DSRLIB';           ! DSR BLISS Library
83      0211 1 %FI
84      0212 1
85      0213 1
86      0214 1 | MACROS:
87      0215 1 |
88      0216 1 |
89      0217 1 | EQUATED SYMBOLS:
90      0218 1 |
91      0219 1
92      0220 1 EXTERNAL LITERAL
93      0221 1 RINTES : UNSIGNED (8);
94      0222 1
95      0223 1 |
96      0224 1 | OWN STORAGE:
97      0225 1 |
98      0226 1 |
99      0227 1 | EXTERNAL REFERENCES:
100     0228 1 |
101     0229 1
102     0230 1 EXTERNAL
103     0231 1 GCA : GCA_DEFINITION,
104     0232 1 IRAC : IRAC_DEFINITION,
105     0233 1 MRA : REF_FIXED_STRING,
106     0234 1 SCA : SCA_DEFINITION,
107     0235 1 TSF : TSF_DEFINITION;
108     0236 1
109     0237 1 EXTERNAL LITERAL                       !Error messages
110     0238 1 RNFLTC;
111     0239 1
112     0240 1 EXTERNAL ROUTINE
113     0241 1 ENDWRD,
114     0242 1 ERMS,
115     0243 1 FCIMRA,
116     0244 1 OUTXHR;

```



```

118 0245 1 GLOBAL ROUTINE ENDCHR (KHAR) : NOVALUE =
119 0246 1
120 0247 1 ++
121 0248 1 FUNCTIONAL DESCRIPTION:
122 0249 1
123 0250 1 ENDCHR takes KHAR and puts it, along with bolding and underlining
124 0251 1 codes, into the MRA. However, the functioning is somewhat more
125 0252 1 complex than that. Basically, KHAR must be held up for one
126 0253 1 character, so that overstriking can be handled correctly.
127 0254 1
128 0255 1 FORMAL PARAMETERS:
129 0256 1
130 0257 1 KHAR is the character to be output when ENDCHR gets called again.
131 0258 1
132 0259 1 IMPLICIT INPUTS: None
133 0260 1
134 0261 1 IMPLICIT OUTPUTS: None
135 0262 1
136 0263 1 ROUTINE VALUE:
137 0264 1 COMPLETION CODES: None
138 0265 1
139 0266 1 SIDE EFFECTS: None
140 0267 1
141 0268 1 --
142 0269 1
143 0270 2 BEGIN
144 0271 2 IF .KHAR EQL RINTES
145 0272 2 THEN
146 0273 2
147 0274 2 IF .SCA_WRD_CPEND EQL RINTES
148 0275 2 THEN
149 0276 2 RETURN
150 0277 2 ELSE
151 0278 3 (0)
152 0279 2 ELSE
153 0280 2 IF .SCA_X_FLAG
154 0281 2 THEN !Send a character to the <INDEX flag> buffer.
155 0282 2 OUTXHR (.KHAR);
156 0283 2
157 0284 2 !Force out previous character
158 0285 2 IF .SCA_WRD_CPEND NEQ RINTES
159 0286 2 THEN
160 0287 3 BEGIN
161 0288 3
162 0289 3 This is where dsr increments some internal counters indicating
163 0290 3 the length of the current word being built. First the sca_wrd_int_l
164 0291 3 is incremented, then the ext_l. The external length is only
165 0292 3 incremented if the character is not included within a passthrough
166 0293 3 sequence or if the character is not printable.
167 0294 3
168 0295 3 FS WCHAR (MRA, .SCA_WRD_CPEND);
169 0296 3 SCA_WRD_INT_L = .SCA_WRD_INT_L + 1;
170 0297 3
171 0298 3 IF (.sca_wrd_cpend AND XX'7F') GEQ XC' '
172 0299 3 AND (.sca_wrd_cpend AND XX'7F') LSS XO'177'
173 U 0300 3 %IF DSRPLUS %THEN
174 U 0301 3 AND .sca_pass EQL false

```



```
175 0302 3 %FI
176 0303      THEN
177 0304      SCA_WRD_EXT_L = .SCA_WRD_EXT_L + 1
178 0305      ELSE
179 0306      IF .SCA_WRD_CPEND EQL %'010'      !Backspace?
180 0307      THEN
181 U 0308      %IF DSRPLUS %THEN
182 U 0309      |
183 U 0310      |   Back up for a backspace if we aren't in the middle
184 U 0311      |   of a passthrough.
185 U 0312      |
186 U 0313      |   BEGIN
187 U 0314      |   IF NOT .SCA_PASS
188 U 0315      |   THEN
189 0316      %FI
190 0317      SCA_WRD_EXT_L = .SCA_WRD_EXT_L - 1;
191 U 0318      %IF DSRPLUS %THEN
192 U 0319      END;
193 0320      %FI
194 0321      END;
195 0322
196 0323      SCA_WRD_CPEND = .KHAR;      !Save character for next time around
197 0324      !Clear single character force indicator.
198 0325      SCA_FRC_CHR = FALSE;
199 0326
200 0327      IF .KHAR EQL RINTES
201 0328      THEN
202 0329      RETURN
203 0330      ELSE
204 0331      SCA_SECT_EMPTY = FALSE;
205 0332
206 0333      !Check for spaces just before the first character of the first word on
207 0334      !the line. This can arise only in .NO FILL mode. The spaces are put
208 0335      !there by the routine OUTXSP, in the module SCANT. It's not necessary to
209 0336      !call FCIMRA in such a case because OUTXSP called it when it detected
210 0337      !the first space, in .NO FILL mode, starting a line. (The special
211 0338      !handling of such leading spaces lets RUNOFF convert lines full of just
212 0339      !spaces, in .NO FILL mode, to just a <CRLF> pair.) In addition,
213 0340      !propagation of change bars, the draft flag, and sequence numbers is
214 0341      !done here. (Logically some of this should be organized otherwise. But
215 0342      !that would make this run slower on -11s.)
216 0343      IF .SCA_FC
217 0344      THEN
218 0345      BEGIN
219 0346      SCA_WRD_BARS = .IRAC_BARS;      !Propagate change bars at start of every word.
220 0347      SCA_WRD_BAR_CHR = .SCA_BAR_CHR;      !...
221 0348
222 0349      SCA_WRD_DRAFT = .GCA_DEBUG_CND;      !Propagate draft character at start of each new line.
223 0350      SCA_WRD_DRAFT_F = .IRAC_DRAFT_FLG;
224 0351      SCA_WRD_ISEQN = .IRAC_ISEQN;      !Set up for /SEQUENCE at start of each new line.
225 0352      SCA_WRD_SEQN_F = .IRAC_SEQN_FLAG;
226 0353      SCA_WRD_IPAGEN = .IRAC_IPAGEN;      !Page number part of sequence number.
227 0354
228 0355      SCA_FC = FALSE;      !Next character is not the first of a word.
229 0356
230 0357      IF (.TSF_INT_HL EQL 0)
231 0358      THEN      !Absolutely nothing in MRA yet? THEN
```



```

232 0359 4      IF (.SCA_WRD_LST_SP EQL 0)      !Leading spaces?
233 0360 3      THEN
234 0361 3      !No leading spaces, so this is just the first character to go into the MRA.
235 0362 3      FCIMRA ()
236 0363 3      ELSE
237 0364 3      !Count leading spaces at the start of the MRA. They are already physically there.
238 0365 4      BEGIN
239 0366 4      TSF_INT_HL = .TSF_INT_HL + .SCA_WRD_LST_SP;
240 0367 4      TSF_EXT_HL = .TSF_EXT_HL + .SCA_WRD_LST_SP;
241 0368 4      SCA_WRD_LST_SP = 0;      !Make sure they don't hang around.
242 0369 3      END;
243 0370 3      END;
244 0371 2
245 0372 2      !In the following check for buffer overflow, "10" comes from the fact that
246 0373 2      !the character plus emphasis coding will require at least 10 characters.
247 0374 2      !This avoids having constant checks whenever something is generated.
248 0375 2      !Having enough space for overstriking too is checked elsewhere.
249 0376 2      IF (.FS_MAXSIZE (MRA) - .FS_LENGTH (MRA)) LSS 10
250 0377 2      THEN
251 0378 3      BEGIN      !Character might not fit in storage
252 0379 3      !The effect of the error handling is that, besides an error message
253 0380 3      !being generated, an end of word is forced. ENDWRD will do
254 0381 3      !additional checking, eventually forcing out the line, and the text
255 0382 3      !causing the problem. When the return is made TSF will be empty,
256 0383 3      !and it will look just like a new word is being started. Then, the
257 0384 3      !remainder of the current line is skipped.
258 0385 3
259 0386 3      UNDECLARE KHAR;      !Don't conflict with the KHAR parameter
260 0387 3      EXTERNAL
261 0388 3      IRA : FIXED_STRING,      !Needed only for error message.
262 0389 3      KHAR;      ! ...
263 0390 3
264 0391 3      ERMS (RNFLTC, .FS_NEXT (IRA), .FS_LENGTH (IRA));
265 0392 3      !End this word, but suppress trailing spaces and justification
266 0393 3      ENDWRD (FALSE, FALSE, FALSE);
267 0394 3      WHILE .KHAR NEQ RINTES DO
268 0395 3      KCNS ();      !Skip to end of offending line.
269 0396 3      RETURN
270 0397 3      END;
271 0398 2
272 0399 2      SCA_FC CASE = FALSE;
273 0400 2      SCA_CONT = TRUE;      !.NO SPACE is legal now.
274 0401 2      SCA_WRD_LC_PNCT = FALSE;      !Turn off 'punctuation', for .PERIOD
275 0402 2
276 0403 2      !Remember bolding and underlining conditions
277 0404 2      SCA_WRD_NBITS = (.SCA_WRD_CNBITS OR .SCA_WRD_NBITS OR .SCA_WRD_ACNBITS);
278 0405 2
279 0406 2      !This select statement depends very much on the arrangement of the bolding
280 0407 2      !and underlining bits in SCA. The reason this code has been written
281 0408 2      !this way is so to make this routine run a bit faster, especially on -11s.
282 0409 2
283 0410 2      SELECT (.SCA_WRD_C_BLDUN OR .SCA_WRD_AC_BLUN) OF
284 0411 2      SET
285 0412 2
286 0413 2      [1,3]:
287 0414 2      !Bolding only or both underlining and bolding.
288 0415 3      BEGIN
```


ENDCHR
V04-000

Process character termination.
Module Level Declarations

E 16
16-Sep-1984 00:23:49
14-Sep-1984 13:06:07

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]ENDCHR.BLI;1

Page 7
(4)

```

: 289      0416      3      SCA WRD INT L = .SCA WRD_INT_L + 3;
: 290      0417      W      FS_WCHAR (MRA, RINTES);
: 291      0418      W      FS_WCHAR (MRA, %C'B');
: 292      0419      W      FS_WCHAR (MRA, .KHAR);
: 293      0420      W      END;
: 294      0421      W
: 295      0422      W      [2,3]:
: 296      0423      W      !Underlining only or both underlining and bolding
: 297      0424      W      BEGIN
: 298      0425      W      SCA WRD INT L = .SCA WRD_INT_L + 3;
: 299      0426      W      FS_WCHAR (MRA, RINTES);
: 300      0427      W      FS_WCHAR (MRA, %C'U');
: 301      0428      W      FS_WCHAR (MRA, %C' ');
: 302      0429      W      END;
: 303      0430      W
: 304      0431      W      TES;
: 305      0432      W
: 306      0433      W      !Set up underlining/bolding as specified by ^& and \&, etc.
: 307      0434      W      SCA_WRD_C_BLDUN = .SCA_BLDUND AND .SCA_DO_BLDUND;
: 308      0435      W      SCA_WRD_ACNBITS = FALSE;
: 309      0436      W      !Additional functions
: 310      0437      W
: 311      0438      W      !Set up case translation rules.
: 312      0439      W      SCA_WRD_FC_UT = .SCA_FC_UT;
: 313      0440      W      SCA_WRD_OC_UT = .SCA_OC_UT;
: 314      0441      W      SCA_WRD_FC_LT = .SCA_FC_LT;
: 315      0442      W      SCA_WRD_OC_LT = .SCA_OC_LT;
: 316      0443      W      SCA_MNWRD_FC_UT = .SCA_MNFC_UT;
: 317      0444      W      SCA_MNWRD_OC_UT = .SCA_MNOC_UT;
: 318      0445      W      SCA_MNWRD_FC_LT = .SCA_MNFC_LT;
: 319      0446      W      SCA_MNWRD_OC_LT = .SCA_MNOC_LT;
:          1      END;

```

!End of ENDCHR

.TITLE ENDCHR Process character termination.
.IDENT \V04-000\

.EXTRN RINTES, GCA, IRAC
.EXTRN MRA, SCA, TSF, RNFLT
.EXTRN ENDWRD, ERMS, FCIMRA
.EXTRN OUTXHR, IRA, KHAR

.PSECT \$CODE\$,NOWRT,2

.ENTRY ENDCHR, Save R2,R3,R4,R5,R6,R7,R8,R9
MOVAB KHAR, R9
MOVAB MRA, R8
MOVZBL #RINTES, R7
MOVAB IRAC, R6
MOVAB IRA+12, R5
MOVAB SCA+280, R4
CLRL R2
CMPL KHAR, R7
BNEQ 1\$
INCL R2
CMPL SCA+260, R7
BNEQ 2\$
RET

```

03FC 00000
59 00000000G EF 9E 00002
58 00000000G EF 9E 00009
57 00000000G 8F 9A 00010
56 00000000G EF 9E 00014
55 00000000G EF 9E 0001B
54 00000000G EF 9E 00022
52 D4 00029
57 04 AC D1 0002B
08 12 0002F
52 D6 00031
57 64 D1 00033
OF 12 00036
04 00038

```

```

: 0245
:
:
:
:
: 0271
:
:
: 0274
:
: 0276

```


ENDCHR
V04-000

Process character termination.
Module Level Declarations

F 16

16-Sep-1984 00:23:49

14-Sep-1984 13:06:07

VAX-11 Bliss-32 V4.0-742

[RUNOFF.SRC]ENDCHR.BLI;1

Page 8
(4)

			0A	84	A4	E9	00039	1\$:	BLBC	SCA+156, 2\$	0280
				04	AC	DD	0003D		PUSHL	KHAR	0282
		00000000G	EF		01	FB	00040		CALLS	#1, OUTXHR	
			57		64	D1	00047	2\$:	CMPL	SCA+280, R7	0285
					2F	13	0004A		BEQL	4\$	
			50		68	D0	0004C		MOVL	MRA, R0	0295
		04	B0		64	90	0004F		MOVB	SCA+280, @4(R0)	
				04	A0	D6	00053		INCL	4(R0)	
				0C	A0	D6	00056		INCL	12(R0)	
				E4	A4	D6	00059		INCL	SCA+252	0296
	20	64	07		00	ED	0005C		CMPZV	#0, #7, SCA+280, #32	0298
					10	19	00061		BLSS	3\$	
0000007F	8F	64	07		00	ED	00063		CMPZV	#0, #7, SCA+280, #127	0299
					05	18	0006C		BGEQ	3\$	
				E8	A4	D6	0006E		INCL	SCA+256	0304
			08		08	11	00071		BRB	4\$	
					64	D1	00073	3\$:	CMPL	SCA+280, #8	0306
					03	12	00076		BNEQ	4\$	
				E8	A4	D7	00078		DECL	SCA+256	0317
		64		04	AC	D0	0007B	4\$:	MOVL	KHAR, SCA+280	0323
				C0	A4	D4	0007F		CLRL	SCA+216	0325
			01		52	E9	00082		BLBC	R2, 5\$	0327
					04	00	00085		RET		
			52	9C	A4	D4	00086	5\$:	CLRL	SCA+180	0331
			00	FF7C	C4	E9	00089		BLBC	SCA+148, 7\$	0343
F8	A4	01			66	F0	0008E		INSV	IRAC, #0, #1, SCA+272	0346
			FC		D4	D0	00094		MOVL	@SCA+136, SCA+276	0347
F0	A4	00000000G	EF		02	EF	0009A		EXTZV	#2, #1, GCA+116, SCA+264	0349
					A6	D0	000A4		MOVL	IRAC+24, SCA+268	0350
			F4	18	A6	D0	000A9		MOVL	IRAC+8, SCA+260	0351
			EC	08	A6	D0	000AE		MOVL	IRAC+4, SCA+284	0352
			04	04	A6	D0	000B3		MOVL	IRAC+12, SCA+288	0353
			08	0C	C4	D4	000B8		CLRL	SCA+148	0355
				FF7C	EF	D0	000BC		MOVL	TSF, R2	0357
		52	00000000G		62	D5	000C3		TSTL	(R2)	
					19	12	000C5		BNEQ	7\$	
			53	34	A4	D0	000C7		MOVL	SCA+332, R3	0359
					09	12	000CB		BNEQ	6\$	
		00000000G	EF		00	FB	000CD		CALLS	#0, FCIMRA	0362
					0A	11	000D4		BRB	7\$	
			62		53	C0	000D6	6\$:	ADDL2	R3, (R2)	0366
		04	A2		53	C0	000D9		ADDL2	R3, 4(R2)	0367
				34	A4	D4	000DD		CLRL	SCA+332	0368
			50		68	D0	000E0	7\$:	MOVL	MRA, R0	0376
51		0C	A0		0A	C1	000E3		ADDL3	#10, 12(R0), R1	
			51	08	A0	D1	000E8		CMPL	8(R0), R1	
					3A	18	000EC		BGEQ	11\$	
					65	DD	000EE		PUSHL	IRA+12	0391
				F8	A5	DD	000F0		PUSHL	IRA+4	
				00000000G	8F	DD	000F3		PUSHL	#RNFLTC	
		00000000G	EF		03	FB	000F9		CALLS	#3, ERMS	0393
					7E	7C	00100		CLRQ	-(SP)	
					7E	D4	00102		CLRL	-(SP)	
		00000000G	EF		03	FB	00104		CALLS	#3, ENDWRD	
			57		69	D1	0010B	8\$:	CMPL	KHAR, R7	0394
					01	12	0010E		BNEQ	9\$	
					04	00	00110		RET		

						65	D5	00111	9\$:	TSTL	IRA+12	0395
						08	14	00113		BGTR	10\$	
			69			57	9A	00115		MOVZBL	R7, KHAR	
			65			01	CE	00118		MNEGL	#1, IRA+12	
						EE	11	0011B		BRB	8\$	
			69			B5	9A	0011D	10\$:	MOVZBL	@IRA+4, KHAR	
					F8	A5	D6	00121		INCL	IRA+4	
					F8	65	D7	00124		DECL	IRA+12	
						E3	11	00126		BRB	8\$	
					B8	A4	D4	00128	11\$:	CLRL	SCA+208	0399
			8C	A4		01	D0	0012B		MOVL	#1, SCA+164	0400
					30	A4	D4	0012F		CLRL	SCA+328	0401
			AC	A4		A4	C9	00132		BISL3	SCA+192, SCA+196, R0	0404
		50		50		A4	C9	00138		BISL3	SCA+200, R0, SCA+192	
52	A8	AC		02		00	EF	0013E		EXTZV	#0, #2, SCA+196, R2	0410
50	BO	BO		02		00	EF	00144		EXTZV	#0, #2, SCA+200, R0	
				52		50	C8	0014A		BISL2	R0, R2	
				01		52	D1	0014D		CMPL	R2, #1	0413
						05	13	00150		BEQL	12\$	
				03		52	D1	00152		CMPL	R2, #3	
						28	12	00155		BNEQ	13\$	
			E4	A4		03	C0	00157	12\$:	ADDL2	#3, SCA+252	0416
				50		68	D0	0015B		MOVL	MRA, R0	0417
				51		A0	9E	0015E		MOVAB	(R0), R1	
			00	B1		57	90	00162		MOVB	R7, @0(R1)	
						61	D6	00166		INCL	(R1)	
					0C	A0	D6	00168		INCL	12(R0)	
			00	B1		42	8F	90 0016B		MOVB	#66, @0(R1)	0418
						61	D6	00170		INCL	(R1)	
					0C	A0	D6	00172		INCL	12(R0)	
			00	B1		04	AC	90 00175		MOVB	KHAR, @0(R1)	0419
						61	D6	0017A		INCL	(R1)	
					0C	A0	D6	0017C		INCL	12(R0)	
				02		52	D1	0017F	13\$:	CMPL	R2, #2	0422
						2C	19	00182		BLSS	14\$	
				03		52	D1	00184		CMPL	R2, #3	
						27	14	00187		BGTR	14\$	
			E4	A4		03	C0	00189		ADDL2	#3, SCA+252	0425
				50		68	D0	0018D		MOVL	MRA, R0	0426
				51		A0	9E	00190		MOVAB	4(R0), R1	
			00	B1		57	90	00194		MOVB	R7, @0(R1)	
						61	D6	00198		INCL	(R1)	
					0C	A0	D6	0019A		INCL	12(R0)	
			00	B1		55	8F	90 0019D		MOVB	#85, @0(R1)	0427
						61	D6	001A2		INCL	(R1)	
					0C	A0	D6	001A4		INCL	12(R0)	
			00	B1		20	90	001A7		MOVB	#32, @0(R1)	0428
						61	D6	001AB		INCL	(R1)	
					0C	A0	D6	001AD		INCL	12(R0)	
				02		00	EF	001B0	14\$:	EXTZV	#0, #2, SCA+152, R0	0434
				02		00	EF	001B6		EXTZV	#0, #2, SCA+168, R1	
				51		51	D2	001BC		MCOML	R1, R1	
				50		51	8A	001BF		BICB2	R1, R0	
				00		50	F0	001C2		INSV	R0, #0, #2, SCA+196	
AC	A4					A4	D4	001C8		CLRL	SCA+200	0435
						C4	D0	001CB		MOVL	SCA, SCA+16	0438
			FEF8	C4		C4	D0	001D2		MOVL	SCA+4, SCA+24	0439
			FF00	C4								

ENDCHR
V04-000

Process character termination.
Module Level Declarations

H 16
16-Sep-1984 00:23:49
14-Sep-1984 13:06:07

VAX-11 Bliss-32 V4.0-742
[RUNOFF.SRC]ENDCHR.BLI;1

Page 10
(4)

FEFC	C4	FEF0	C4	DO	001D9
FF04	C4	FEF4	C4	DO	001E0
FF28	C4	FF18	C4	DO	001E7
FF30	C4	FF1C	C4	DO	001EE
FF2C	C4	FF20	C4	DO	001F5
FF34	C4	FF24	C4	DO	001FC
				04	00203

MOVL	SCA+8,	SCA+20
MOVL	SCA+12,	SCA+28
MOVL	SCA+48,	SCA+64
MOVL	SCA+52,	SCA+72
MOVL	SCA+56,	SCA+68
MOVL	SCA+60,	SCA+76
RET		

: 0440
: 0441
: 0442
: 0443
: 0444
: 0445
: 0446

; Routine Size: 516 bytes, Routine Base: \$CODE\$ + 0000

: 320 0447 1
: 321 0448 1 END
: 322 0449 0 ELUDOM

!End of module

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	516	NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
\$255\$DUA28:[SYSLIB]XPORT.L32;1	590	0	0	252	00:00.1
\$255\$DUA28:[RUNOFF.SRC]DSRLIB.L32;1	1248	74	5	86	00:00.3

COMMAND QUALIFIERS

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:ENDCHR/OBJ=OBJ\$:ENDCHR MSRC\$:ENDCHR/UPDATE=(ENH\$:ENDCHR)

; Size: 516 code + 0 data bytes
; Run Time: 00:12.4
; Elapsed Time: 00:33.9
; Lines/CPU Min: 2172
; Lexemes/CPU-Min: 22040
; Memory Used: 149 pages
; Compilation Complete

0340

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY